2022

CAB230 Assignment 1 Client Side

CAB230

Volcano API – Client Side Application

Harrison Kelly

N9986189

5/29/2022

Contents

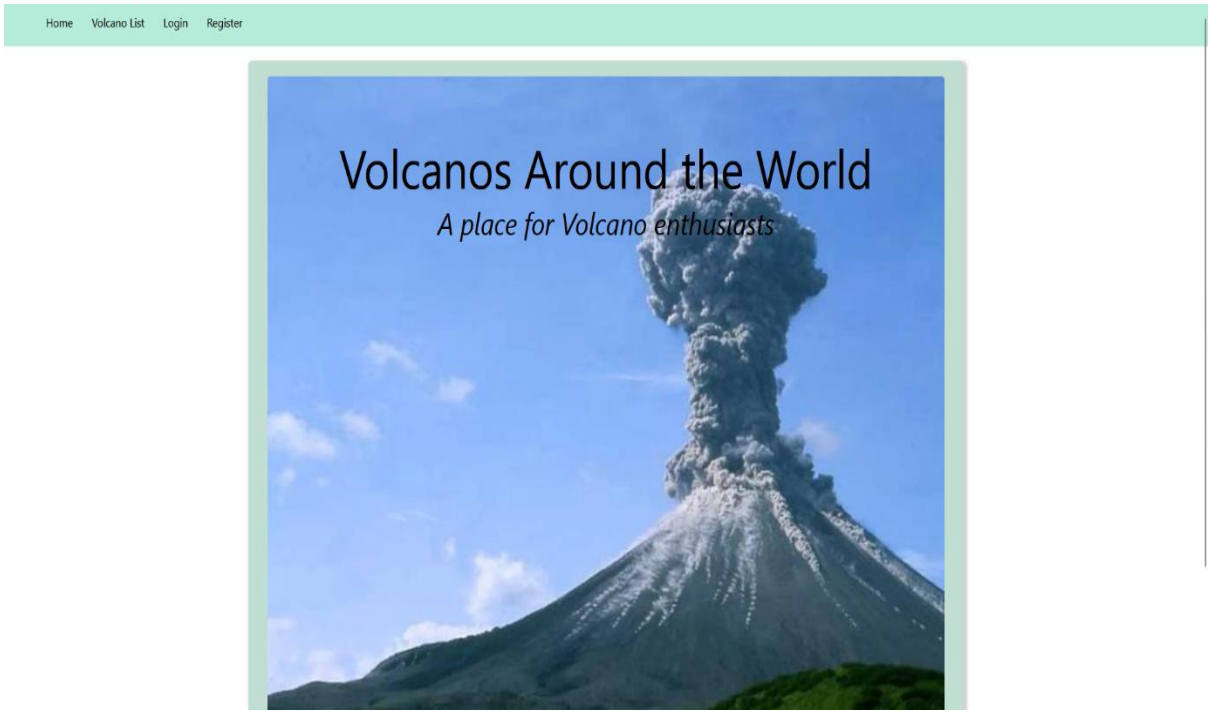
Introduction

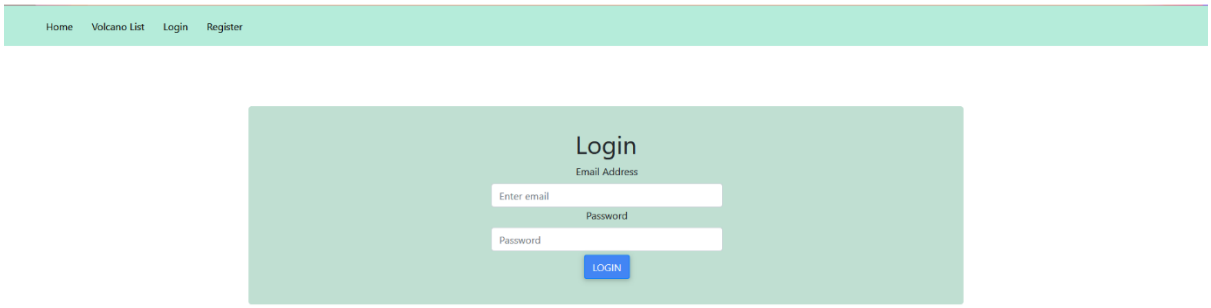
Purpose & description

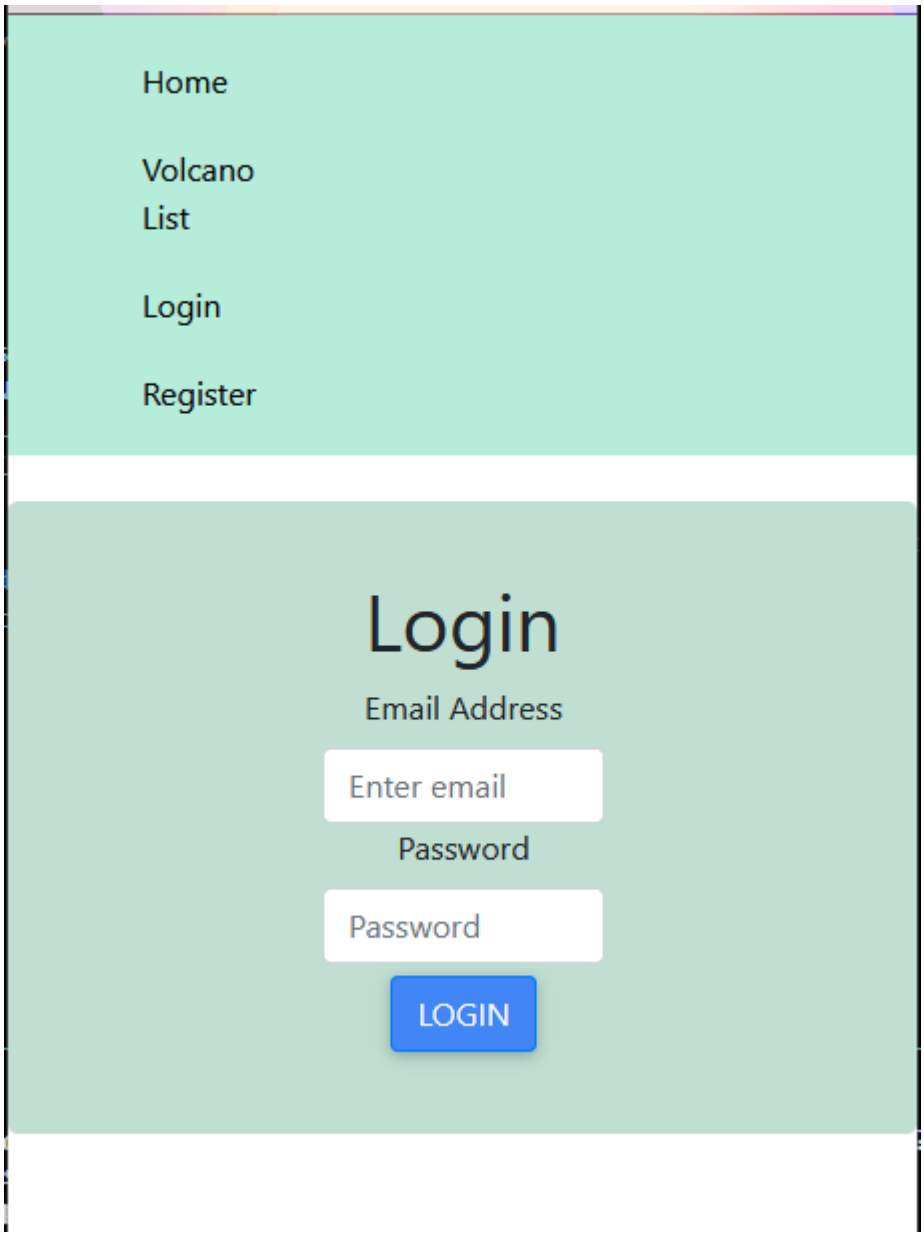
The finished web application is designed to provide the ability for the user to register an account and log in to their account. When not logged in the user can navigate through the website freely and view volcano data from different countries in a table format with filters to refine their search results. The filterable options include the country name and the volcanos surrounding a population within varied distances. When the user clicks on a specific volcano in the table they are routed to a page that visually displays the location of the volcano on a map and some basic volcano data on the side. When the user is logged in, a population density chart is displayed to show the population density surrounding that specific volcano.

The modules used in this application were standard in terms of complexity. However, the “react-chartjs-2” module used for the population density bar chart was formatted quite smoothly with hover tags showing the exact number of people within each specified distance of a volcano. In addition, when the page loads there is an animation showing the bars rising in the bar chart.

The coloring of the application was designed to incorporate soft colors to not overwhelm the user. Screenshots have been provided below to show the application design. One thing to keep an eye out for is the fact that all the pages collapse well when decreasing the window size, however, the navbar doesn’t collapse very neatly




Home
Volcano List
Login
Register
Login
Email Address
Enter email
Password
Password
LOGIN


Home
Volcano List
Login
Register
Login
Email Address
Enter email
Password
Password
LOGIN

Completeness and Limitations

Most of the functionality for the website is provided successfully when looking at the guidelines. The login functionality works well, it creates a user and stores the valid token in local storage. Upon login, the navbar changes to incorporate a logout option. The logout option works and removes the token from local storage which then initiates a visual and functional change in the navbar back to its original form. The search and filter functionality works well and displays the correct information in the table, as well as displaying the correct data on a new page when a row is clicked. The map and location implementation works as it should when logged in and so does the population density chart. When logged out, the map and basic volcano data are still available but the bar chart is not as it requires a valid JWT token for authorization. Unfortunately, when navigating back to the page containing the table, the previous data is lost and has to be entered again.

All in all, I think the website works well, however, when the current user's token expires and the website does not log them out automatically which would have been a good addition to functionality. Also, when searching data in the volcano lists page and clicking on a row to view the data, the data is lost when navigating back to the search page which unfortunately becomes tedious when you want to explore the same data. However, when considering the functionality of the website as a whole, it achieves all the requirements that were asked besides a few persistence issues. When critiquing the website, the previous statement should be regarded.

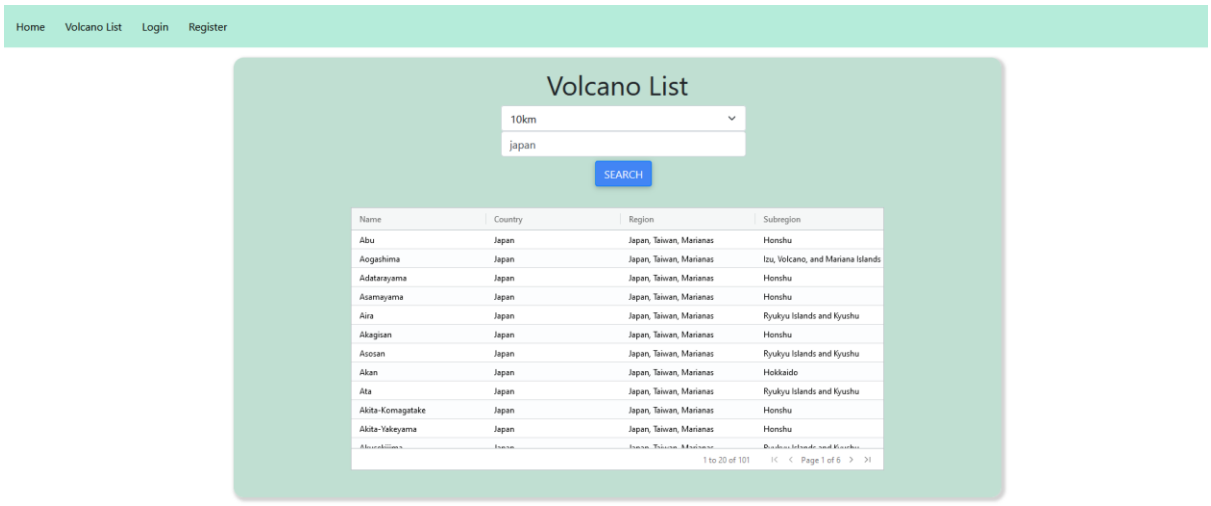
Use of End Points

/countries

This API endpoint was not used in the application as most of the functionality was achieved without it. However, it would have been good to have it implemented as an auto-complete function in the search bar.

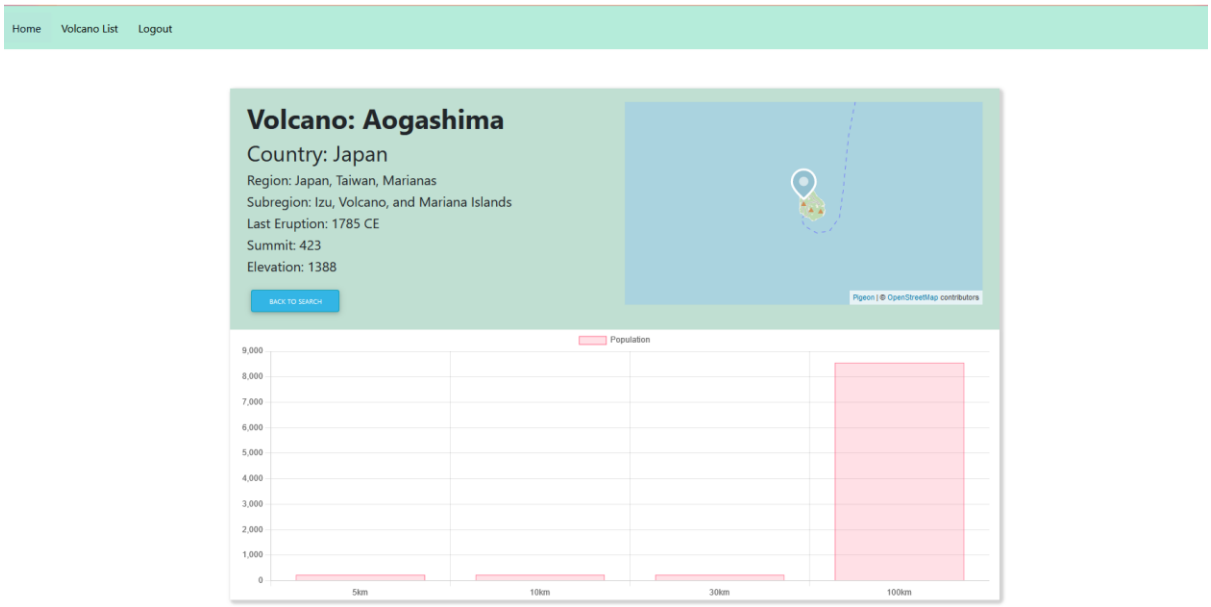
/volcanoes

This API endpoint is used to fetch all the volcanoes depending on what filters are used in the search functionality. When the filters are added they are parsed into the query to fetch the data.



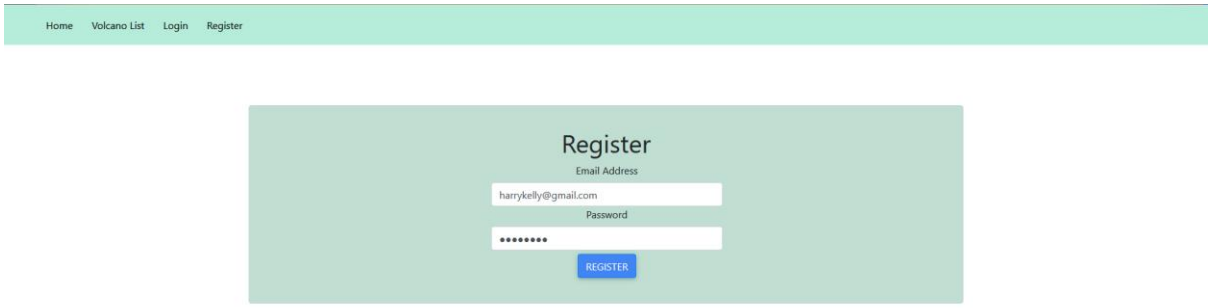
/volcano/{id}

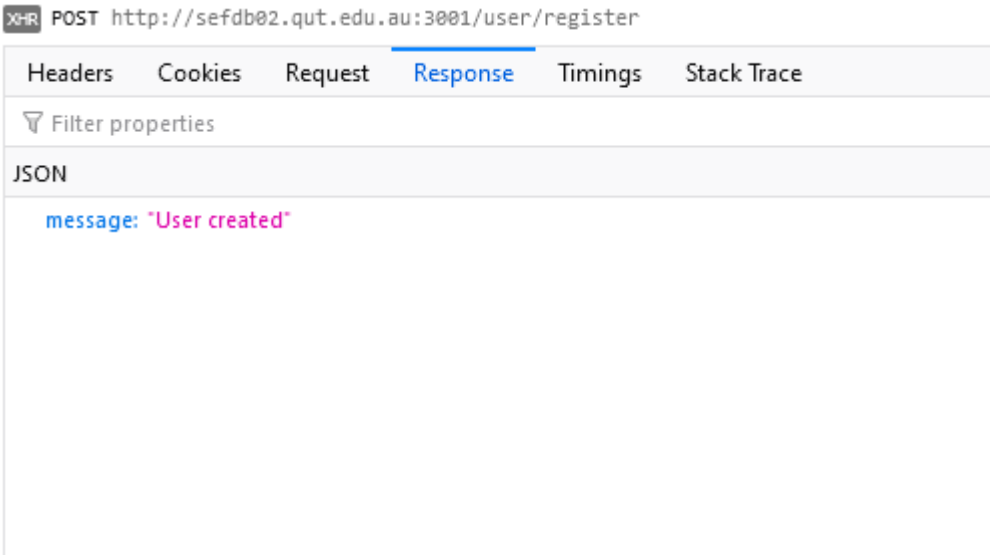
This API is used to fetch the data associated with a specific volcano using an authorized GET request. The id is parsed into the query to obtain the data that allows the user to view the location on the map, and the population data on the bar chart.



/user/register

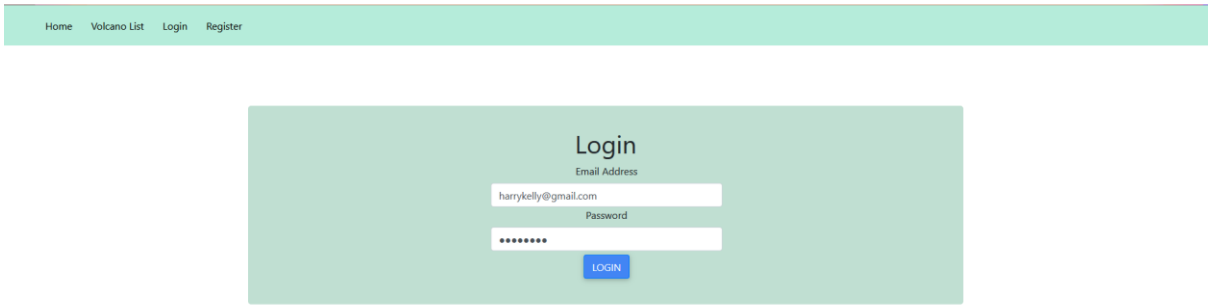
This API is used to make a POST request using the fetch function. The POST request will take the email and password data input by the user and store it in local storage to be used to authorize a user login.





/user/login

This API is used to authorize the user via a POST request using the fetch function. When existing credentials are entered a POST request is sent to the API confirming that those credentials exist and returning a valid token that is stored in local storage.



Modules Used

Ag-grid-react

This module is used to aggregate and format data from the /volcanoes API in a table format. The module has a specific functionality where a data structure must be defined, then using that data structure in a fetch function it will return the data in column and row form within the <AgGridReact> tag.

https://www.ag-grid.com/react-grid/

pigeon-maps

This module utilizes the latitude and longitude data obtained from the /volcano{id} API to visually represent the location of a volcano on an interactive map. The module comes with the <Map> and <Marker> tags to show the location and place a marker on that location. The module requires the JSON latitude and longitude data to be parsed into the <Map> tag as a float to work.

https://www.npmjs.com/package/pigeon-maps

react-chartjs-2

This module utilizes data obtained from the /volcano{id} API via an authorized fetch request. The population density is inserted into the modules <Bar> tag to visually represent the data obtained from the API in a bar chart

https://www.npmjs.com/package/react-chartjs-2

reactstrap

This module is used to format the pages in the web application in grid form. Using the modules <Container>, <Col>, and <Row> tags this module helps with formatting page contents easily and helps with collapsing HTML components when reducing screen width and height.

https://www.npmjs.com/package/reactstrap

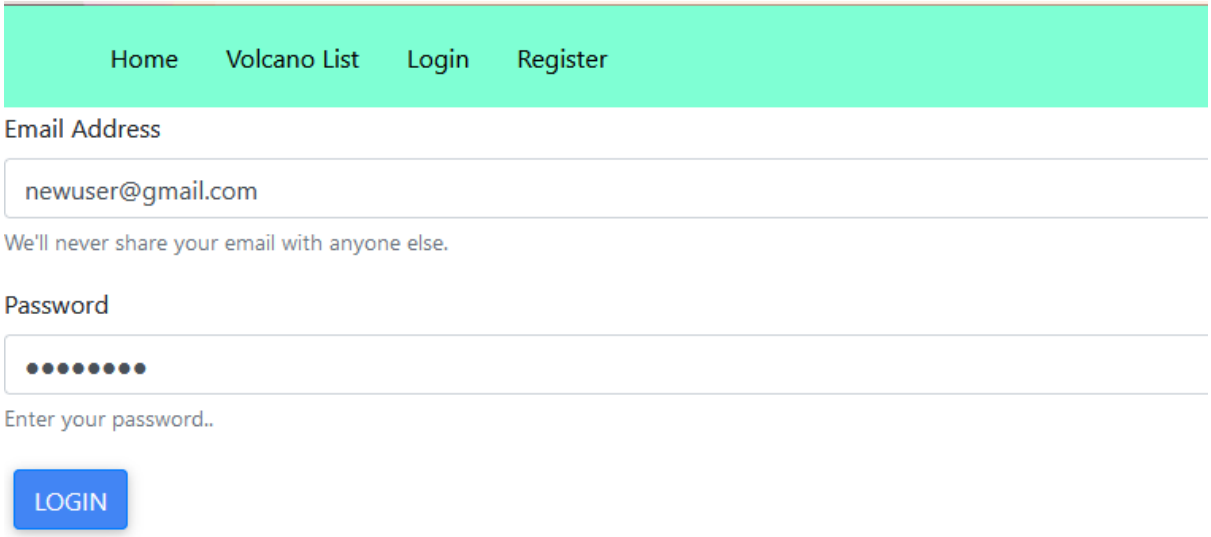
Application Design

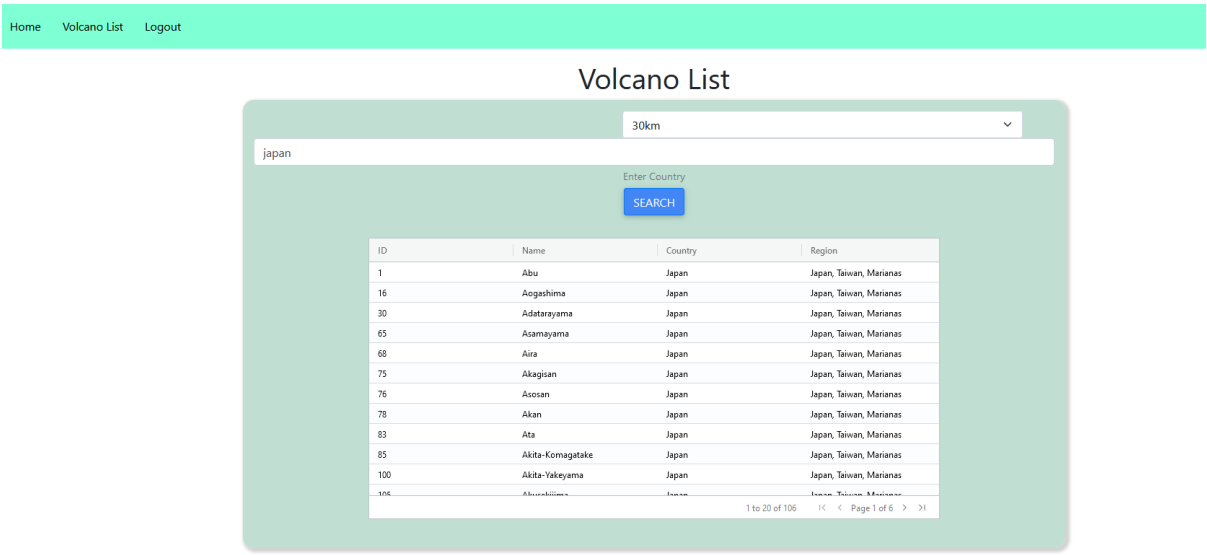
Navigation and Layout

The website was designed to provide an obvious and familiar layout for the user. The navbar is simple with a Home, Register, Login, and Volcano List buttons. It is intended for the user to immediately understand where they are and how where to go. The landing page is a simple picture with text, the user is encouraged to start clicking on the navbar items when arriving on the main page.

The register page is a simple layout with email and password bootstrap forms. The page is designed to take the user to the login page immediately after the user has registered an account. The login page has the same format as the register page, if the user enters in the wrong or incomplete credentials, an alert will pop up telling the user. When successfully logged in, the navbar will change to exclude the register button and the user will be redirected back to the home page. The navbar will have the implemented logout button and functionality, when the button is clicked, the user will be redirected to the homepage from wherever they are currently. The Volcano List page can be accessed whether the user is logged in or out, and they can search and filter volcano data freely. However, when clicking a row on the table the user can only see location data and basic volcano data.

In the design process, different color palettes were used and the formatting of the search and input bars on the Volcano List, Register, and Login pages were wonky. Below are some screenshots of the initial design and format.





Usability and Quality of Design

The layout of the website was intended to be clean, collapsible, and generally well-formatted. The website is logically laid out and intended to be as simple as possible. The homepage unfortunately isn't very purposeful, the picture used is too large and the text on the picture is quite hard to see. To incorporate a more simple bootstrap jumbotron with a smaller picture would have been preferred and nicer on the eyes. In addition, it would have been better to add some navigation options to the picture. The navbar however stays simple, consistent, and usable when navigating through the website.

The Login and Register pages achieve their purpose. However, having a bootstrap card with shadow and some depth would have solidified the look of both pages and highlighted their purpose a lot better.

The Volcano List page containing the table data is well designed apart from the search and selection bars above the table. This page functions as it should, however, the search bars above the table are stacked, which doesn't make the page look professional. Most websites have their search bars side by side and in line with the edges of whatever is beneath it

The design of the website is visually consistent on all pages. The buttons on the Volcano List Volcano Data, login, and Register pages are a bit too bright and should have been given softer colors to fit the color palette of the website. However, considering the usability of the website it functions fine. As mentioned before, unfortunately, the searched data does not show up on the Volcano List page when a user navigates back from the Volcano Data page.

Accessibility

- **Provide a text equivalent for every non-text element – alternatives to images, symbols, scripts, graphical buttons, sounds, audio and video files:**
  Text equivalents were provided for the ag-grid-react table, the landing page image, and the population density charts. Since there was not a lot of content that required text equivalents

this requirement was mostly achieved. Input boxes were also given text alternatives, however, the buttons had ‘types’ instead of text equivalents which could hinder the website's code readability and possibly cause confusion.

Ensure that all information conveyed with color is also available without color, for example from context or markup.:
The website's contents along with color make use of text to convey the specific purpose of a page for the most part. The Login and Register pages have text placeholders that tell the user what each textbox is used for along with a large container title that outlines the purpose of both forms. On the Volcano List page, text placeholders are also provided and muted out to highlight what the user needs to enter to get results. However, since the country input field is a required search parameter, a red star and a small label should be placed next to it. This would highlight to the user that the country search box cannot be empty and a valid search query must be entered.

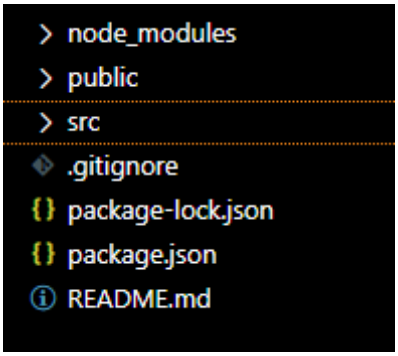
The Volcano Data page provides the user with self-explanatory information about the volcano along with a map location. The map and graph however should have been labeled to highlight their purpose and interactivity. When hovering the cursor over that bard on the graph an accurate population number is given which helps the user understand the content of the graph. In addition, the user should have a visual representation showing them that they are logged in on the navbar. Because this is not added, it can hinder the users understanding if they are logged in or not.

- **Organize documents so they may be read without style sheets. For example, when an HTML document is rendered without associated style sheets, it must still be possible to read the document:**
  When the user deletes the App.css style sheet in the source folder, the website functions as it should. However, the navbar options clump together which makes it hard to understand what option to click or if the user is logged in or out. The <header> parent element for the navbar and the <main> element for the content below the navbar are defined on each page which makes the website readable even without a stylesheet.
- **Ensure that text equivalents are updated when dynamic content changes.**
  The most vital text equivalent is updated when a user has logged in and logged out. The navbar updates accordingly. However, there is no characteristic of the webpage that indicates whether the user is looking at authorized or unauthorized data, especially on the Volcano List and Volcano Data page. This hinders the functionality of the website as it is not obvious what content they are viewing.
- **Avoid causing the screen to flicker.**
  There is no flickering when navigating through the website. There is a flicker on the Home page when the user logs out. A small glimpse of the Home page is rendered and then re-rendered again.
- **Use the clearest and simplest language appropriate for a site's content:**
  Each link and page is clearly defined and contains a title signifying the purpose of the page. Each text field and button is labeled that highlights its purpose as well.
- **For tables, identify row and column headers – clearly differentiated from the data.**
  On the Volcano List page, the ag-grid-react module was used to display the table. Columns in the table are labeled correctly and have the correct data displayed in them on each row. The format of the table is easily readable and uncomplicated.

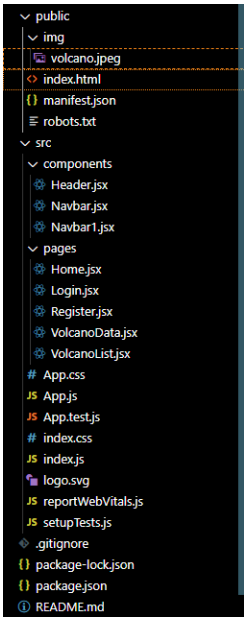
Technical Description

Architecture

The node_modules folder contains all the npm react modules that require the application to work such as the ag-grid-react module. The public folder contains the index.html page which is the page template for all the pages in the application, it also contains a subfolder called img which contains the image for the Home page. The src folder contains two subfolders called pages and components. The pages folder contains all the code for each page. The components folder contains the code for the two different navbar states and the header code which is vital for defining where the navbar and main content are placed.



As seen below the source folder contains the components folder and the pages folder. The NavBar.jsx component contains the code for the default navbar which is the logged-in state, and the NavBar1.jsx component contains the code for the navbar in the logged-in state. Both these components contain <Link> elements to all the different pages and are imported into the Head.jsx component. The Head.jsx component is responsible for initializing the navbar when imported into App.js which runs the application. The pages folder contains all the code for each separate page, each page renders different content under the <main> HTML tag to be placed under the navbar and is exported and imported into App.js. The App.js file contains all the <Route> tags and imported functions to render each page when navigating to the pages. Each element in each component is given a className so that the App.css file when imported into the App.js file styles all the rendered HTML provided by each page. Code snippets and the file structure has been provided below.




```jsx
import './App.css';
import {BrowserRouter, Routes, Route} from "react-router-dom";
import 'bootstrap/dist/css/bootstrap.min.css';
import "ag-grid-community/dist/styles/ag-grid.css";
import "ag-grid-community/dist/styles/ag-theme-balham.css";
import Header from './components/Header';
import Home from './pages/Home';
import VolcanoList from './pages/VolcanoList';
import Login from './pages/Login';
import Register from './pages/Register';
import VolcDataDisplay from './pages/VolcanoData';



export default function App() {
  return (
    <BrowserRouter>
      <div className="App">
        <Header/>
        <Routes>
          <Route path="/" element={<Home />} />
          <Route path="/volcanolist" element={<VolcanoList/>}/>
          <Route path="/Login" element={<Login/>}/>
          <Route path="/Register" element={<Register/>}/>
          <Route path="/volcanodata" element={<VolcDataDisplay/>}/>
        </Routes>
      </div>
    </BrowserRouter>
  );
}
```

```jsx
import React from "react";
import { useState, useEffect } from "react";
import NavBarUser from "./Navbar";
import NavBar from "./Navbar1";

const Header = () => {
    const [userData, setUserData] = useState();
    useEffect(() => {
        setUserData(localStorage.getItem("token"));
    }, [localStorage.getItem("token")]);

    return(
        <header>
            <>
                {userData ? <NavBar/>  : <NavBarUser/>}
            </>
        </header>
    );
}

export default Header;
```

Test plan

Task	Expected Outcome	Result	Screenshots (Appendix A)
Register user	Successfully registered account confirmed by JSON response	PASS	Appendix 1(a) Appendix 1(b)
Login	Successful login in with newly registered credentials	PASS	Appendix 2(a) Appendix 2(b)
Logout	Successful logout and JWT token removed from storage	PASS	Appendix 3(a) Appendix 3(b)
Navbar changes on login	Navbar successfully changes when a user logs in	PASS	Appendix 4(a)
Navbar changes on logout	Navbar successfully changes when a user logs out	PASS	Appendix 5(a)
Searching a country	Searching a country without a specified distance returns all volcanoes in the table	PASS	Appendix 6(a) Appendix 6(b)
Searching a country with distance	Searching a country with a specified distance filters results accurately	PASS	Appendix 7(a) Appendix 7(b)
Clicking the table row when logged out	When clicking a row on the table only basic volcano data and map data are shown	PASS	Appendix 8(a) Appendix 8(b)
Clicking the table row when logged in	When clicking a row on the table all data is shown including the population density chart	PASS	Appendix 9(a) Appendix 9(b)

Difficulties / Exclusions / unresolved & persistent errors /

During application design, there were a few roadblocks such as rendering the table. It was difficult to render data when a distance wasn't selected. When clicking the search button, a useEffect hook and multiple useState hooks were used to store and format data in an array from one function containing a fetch request that contained the API URL with the country and id included. To solve this problem, two functions with two separate fetch requests were created. One with the URL containing the id and country, and another with the URL containing only the country. Two useState hooks were also used to set the country name and the country id. A conditional statement was put in the first function to check if there was an id in the URL, and if there wasn't the function that only used the country for the URL was called instead. The function holding the conditional statement was called a submit handler function. In addition, it is also called a useEffect hook that contained the country and id as dependencies. In the end, it worked, a screenshot of the final function is provided below.

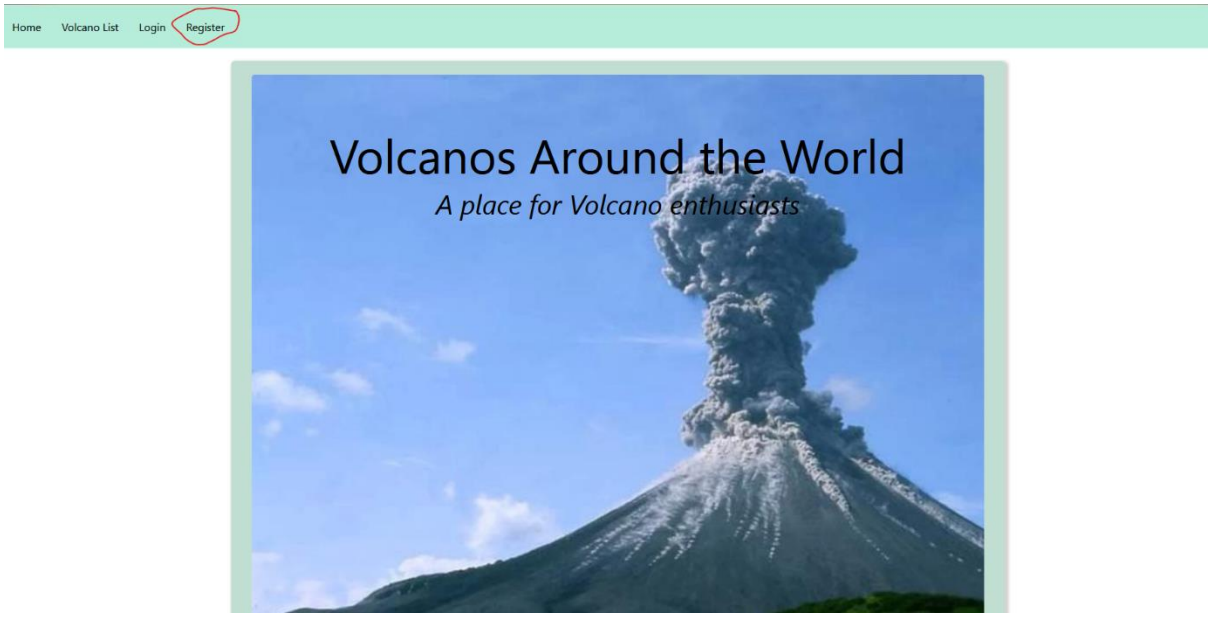
```
function SettingRowData(event){
    event.preventDefault();
    setChosenCountry(chosenCountry)
    setChosenDistance(chosenDistance)
    getTableData(chosenCountry, chosenDistance)
    .then(rowData => setRowData(rowData))
    useEffect(() => {

    }, [chosenCountry, chosenDistance])
}
```

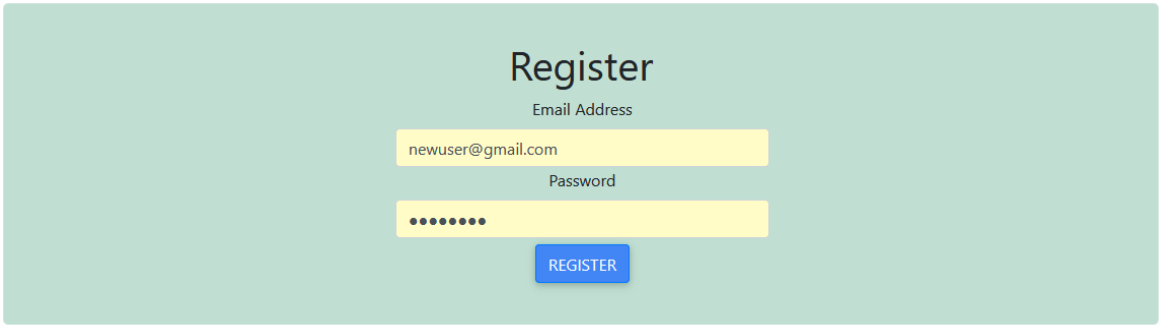
There were also persistency errors when navigating back to the Volcano List page from the Volcano Data page. Data did not persist in the table when navigating back. In addition, there is no mechanism to log the user out after their token expires.

User guide

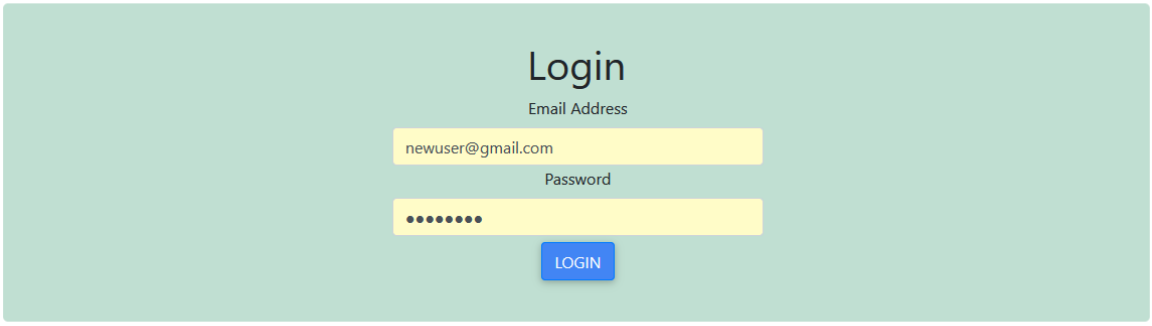
Upon starting up the application the user is represented with a landing page. Click the “Register” link on the navbar to register an account:



Then enter in your desired credentials:

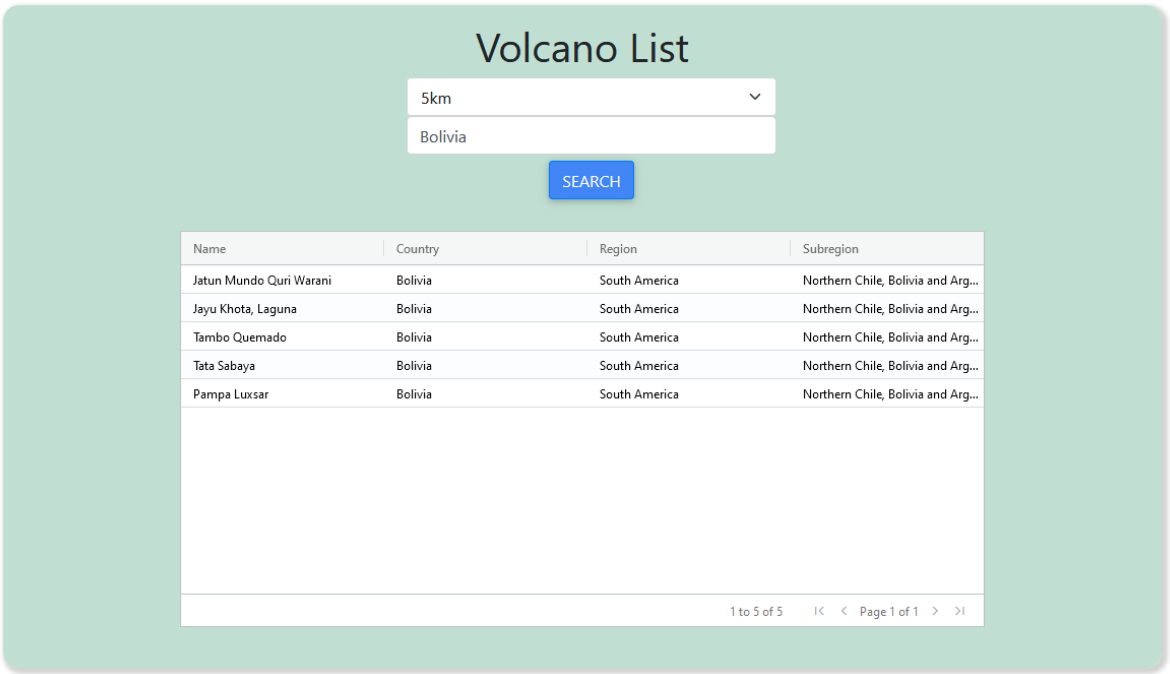


You will then be redirected to the login page. Re-enter your credentials and click login;

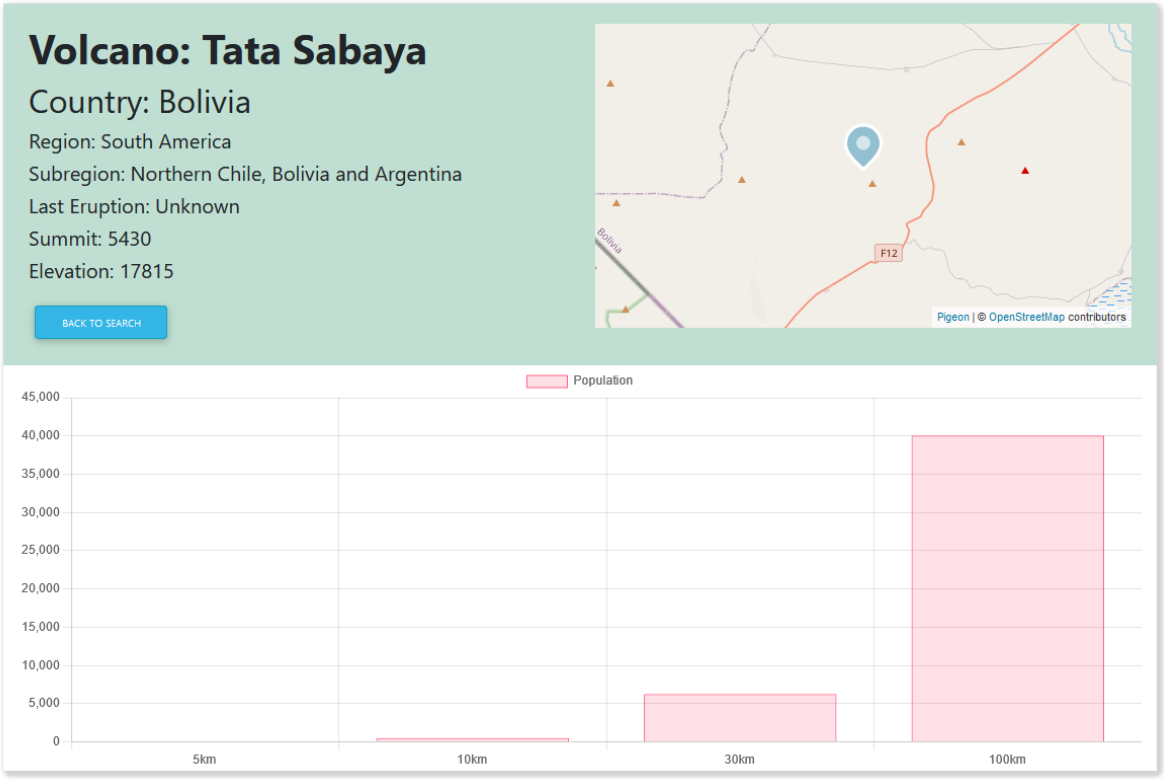


You will then be redirected back to the landing page. Now click on the “Volcano List” link on the navbar:

On the “Volcano List” page enter a country and click search. Choose a distance to refine your search if you wish. When presented with the table of data, click a volcano of your choosing:



You will then be directed to a page containing the data of the volcano you chose. A graph of the population density and a map of the location is also presented. Not when you are logged out, you can search volcano and see the basic data and map location, but not the graph. You may hover your cursor over the bars in the graph to see a specific population number and you can interact with the map.



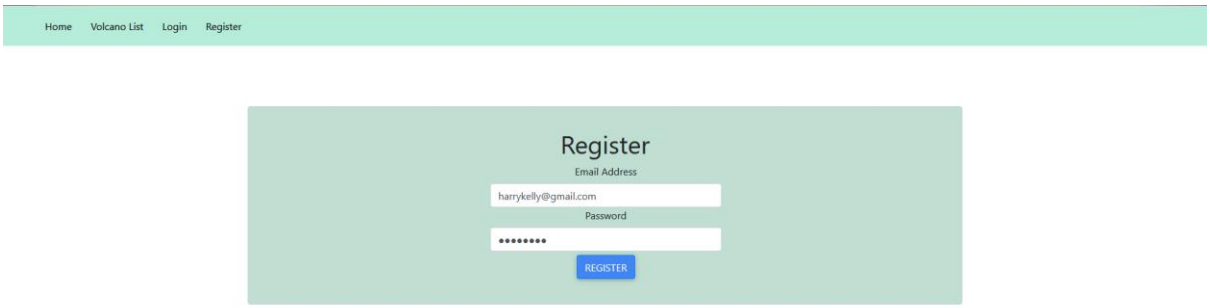
References

(For image used in code)

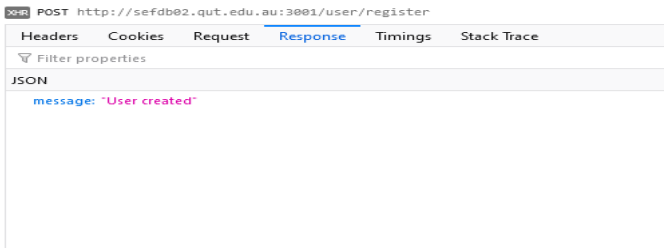
How Much CO2 Does A Single Volcano Emit?.2017. "Title of webpage." Accessed May 27, 2022. https://www.forbes.com/sites/startswithabang/2017/06/06/how-much-co2-does-a-single-volcano-emit/?sh=4096af0a5cbf

Appendix A

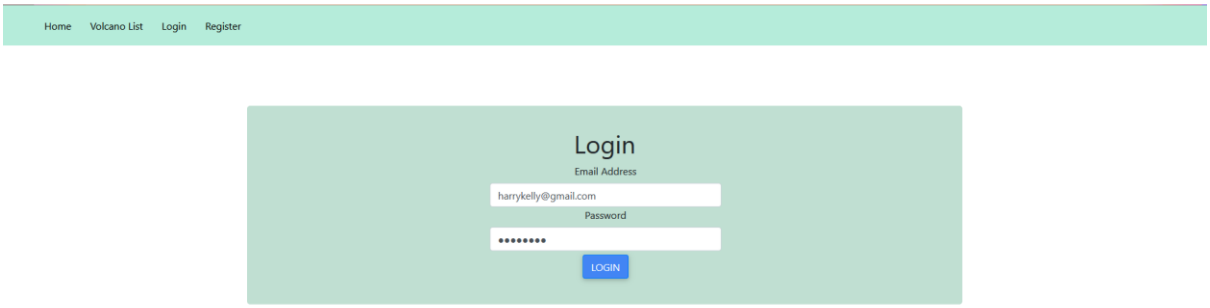
1(a):



1(b):



2(a):



2(b):

3(a):

email	undefined
password	undefined
token	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFpbCI6Im1pa2VAZ21haWwuY29tIiwiZXhwIjoxNjUzOTEyOTUzLCJpYXQiOjE2NTM4MjY1NTN9.pOyoC8SrzmjJ43PdwU95Ite-Bgts4Lp6gD8FL3ULW3M

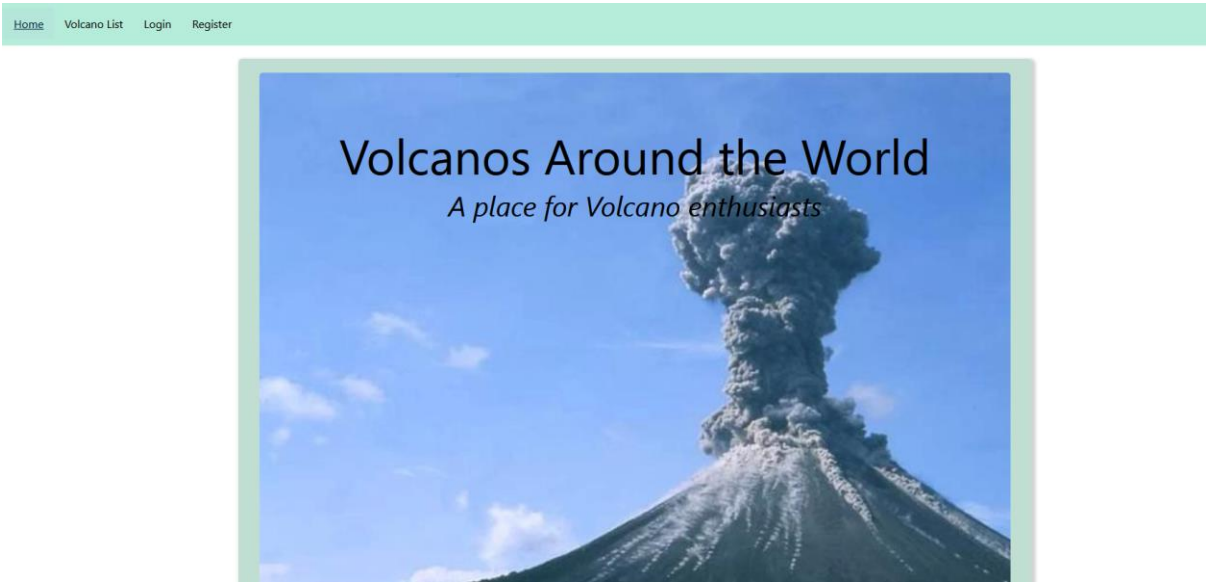
3(b):

Key	Value
email	undefined
password	undefined

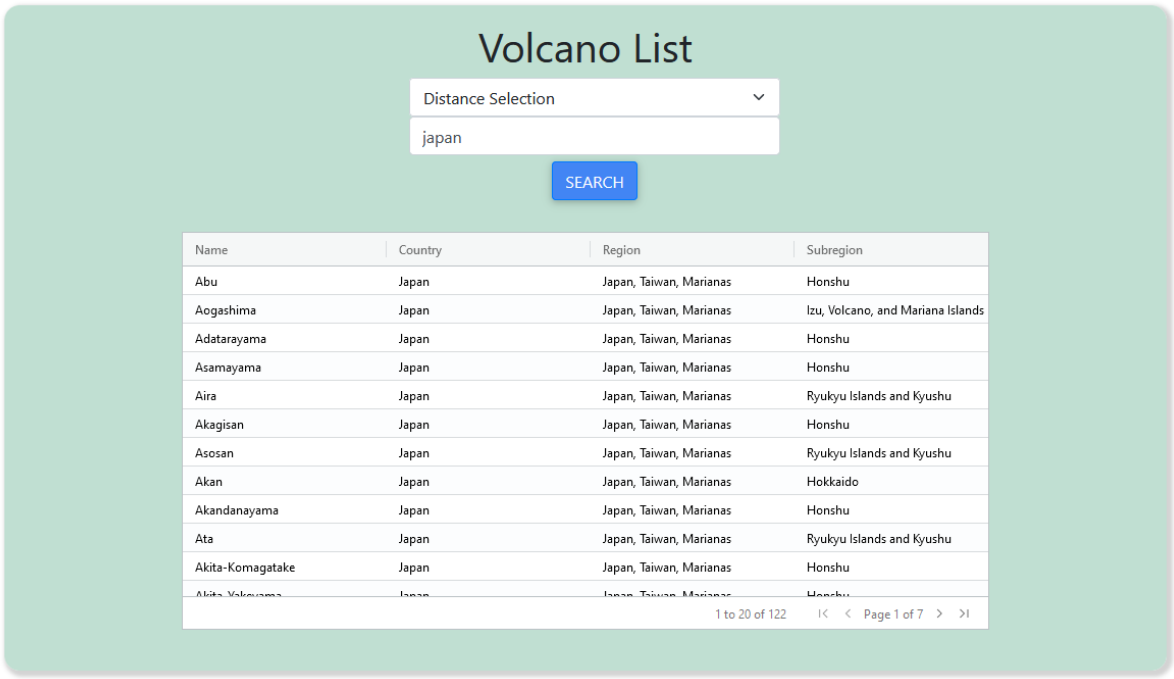
4(a):


Home
Volcano List
Logout
Volcanos Around the World
A place for Volcano enthusiasts

5(a):


Home
Volcano List
Login
Register
Volcanos Around the World
A place for Volcano enthusiasts

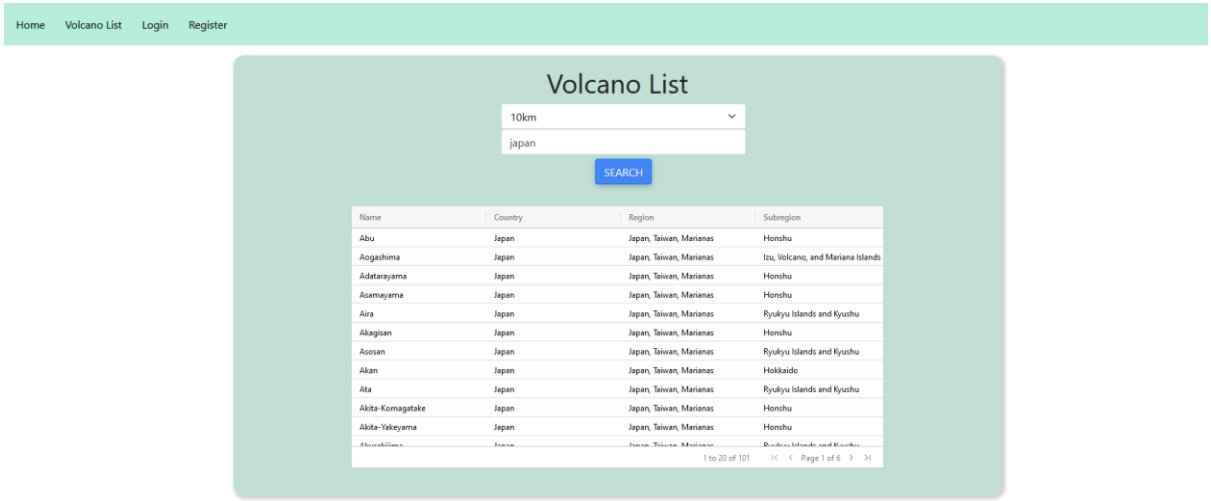
6(a):


Volcano List
Distance Selection
japan
SEARCH
Name
Country
Region
Subregion
Abu
Japan
Japan, Taiwan, Marianas
Honshu
Aogashima
Japan
Japan, Taiwan, Marianas
Izu, Volcano, and Mariana Islands
Adatarayama
Japan
Japan, Taiwan, Marianas
Honshu
Asamayama
Japan
Japan, Taiwan, Marianas
Honshu
Aira
Japan
Japan, Taiwan, Marianas
Ryukyu Islands and Kyushu
Akagisan
Japan
Japan, Taiwan, Marianas
Honshu
Asosan
Japan
Japan, Taiwan, Marianas
Ryukyu Islands and Kyushu
Akan
Japan
Japan, Taiwan, Marianas
Hokkaido
Akandanayama
Japan
Japan, Taiwan, Marianas
Honshu
Ata
Japan
Japan, Taiwan, Marianas
Ryukyu Islands and Kyushu
Akita-Komagatake
Japan
Japan, Taiwan, Marianas
Honshu
1 to 20 of 122
Page 1 of 7

6(b):


XHR GET http://sefdb02.qut.edu.au:3001/volcanoes?country=Japan
Headers
Cookies
Request
Response
Timings
Stack Trace
Filter properties
JSON
0: Object { id: 1, name: "Abu", country: "Japan", … }
1: Object { id: 16, name: "Aogashima", country: "Japan", … }
2: Object { id: 30, name: "Adatarayama", country: "Japan", … }
3: Object { id: 65, name: "Asamayama", country: "Japan", … }
4: Object { id: 68, name: "Aira", country: "Japan", … }
5: Object { id: 75, name: "Akagisan", country: "Japan", … }
6: Object { id: 76, name: "Asosan", country: "Japan", … }
7: Object { id: 78, name: "Akan", country: "Japan", … }

7(a):


Home
Volcano List
Login
Register
Volcano List
10km
japan
SEARCH
Name
Country
Region
Subregion
Abu
Japan
Japan, Taiwan, Marianas
Honshu
Aogashima
Japan
Japan, Taiwan, Marianas
Izu, Volcano, and Mariana Islands
Adatarayama
Japan
Japan, Taiwan, Marianas
Honshu
Asamayama
Japan
Japan, Taiwan, Marianas
Honshu
Aira
Japan
Japan, Taiwan, Marianas
Ryukyu Islands and Kyushu
Akagisan
Japan
Japan, Taiwan, Marianas
Honshu
Asosan
Japan
Japan, Taiwan, Marianas
Ryukyu Islands and Kyushu
Akan
Japan
Japan, Taiwan, Marianas
Hokkaido
Ata
Japan
Japan, Taiwan, Marianas
Ryukyu Islands and Kyushu
Akita-Komagatake
Japan
Japan, Taiwan, Marianas
Honshu
Akita-Yakeyama
Japan
Japan, Taiwan, Marianas
Honshu
1 to 20 of 101
Page 1 of 6

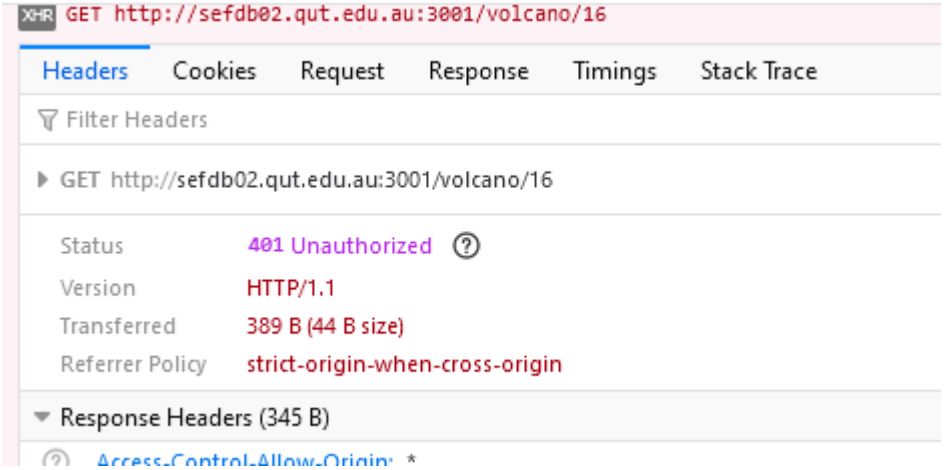
7(b):


XHR GET http://sefdb02.qut.edu.au:3001/volcanoes?country=japan&populatedWithin=10km
Headers
Cookies
Request
Response
Timings
Stack Trace
Filter properties
JSON
0: Object { id: 1, name: "Abu", country: "Japan", ... }
1: Object { id: 16, name: "Aogashima", country: "Japan", ... }
2: Object { id: 30, name: "Adatarayama", country: "Japan", ... }
3: Object { id: 65, name: "Asamayama", country: "Japan", ... }
4: Object { id: 68, name: "Aira", country: "Japan", ... }
5: Object { id: 75, name: "Akagisan", country: "Japan", ... }
6: Object { id: 76, name: "Asosan", country: "Japan", ... }
7: Object { id: 78, name: "Akan", country: "Japan", ... }
8: Object { id: 83, name: "Ata", country: "Japan", ... }
9: Object { id: 85, name: "Akita-Komagatake", country: "Japan", ... }

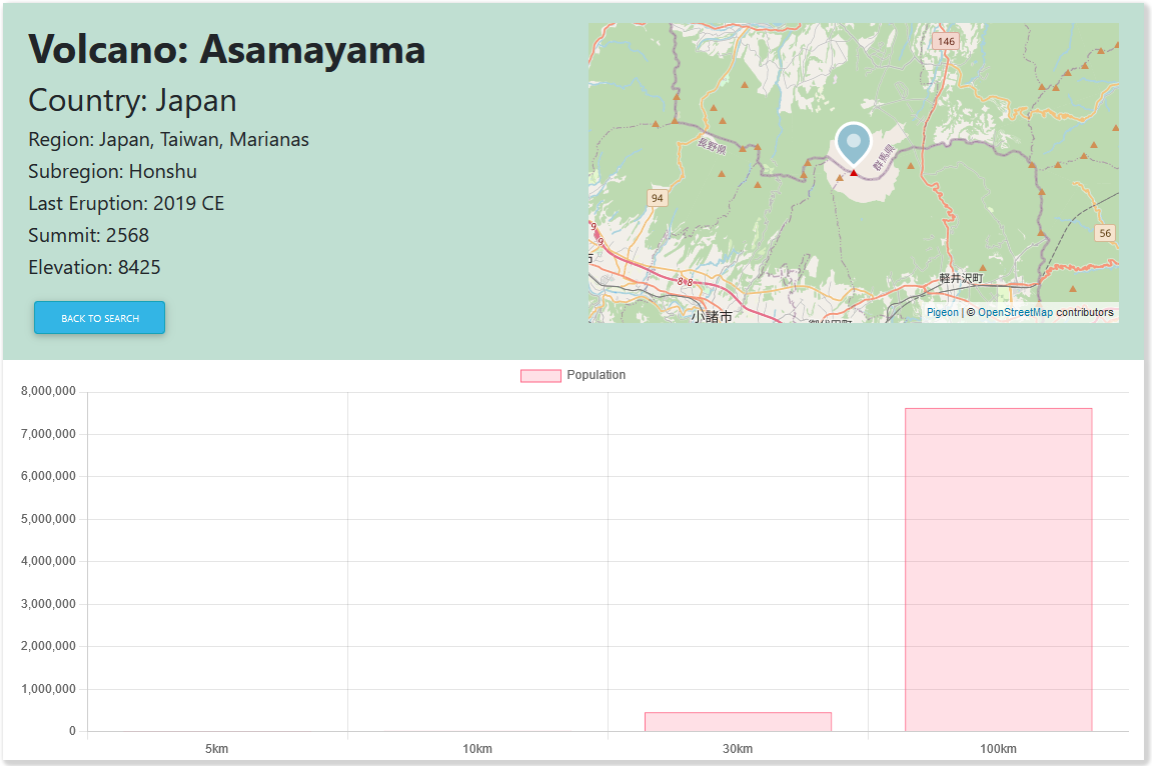
8(a):


Volcano: Adatarayama
Country: Japan
Region: Japan, Taiwan, Marianas
Subregion: Honshu
Last Eruption: 1996 CE
Summit: 1728
Elevation: 5669
BACK TO SEARCH
115
E4
52
459
115
459
Pigeon | © OpenStreetMap contributors

8(b):


XHR GET http://sefdb02.qut.edu.au:3001/volcano/16
Headers
Cookies
Request
Response
Timings
Stack Trace
Filter Headers
GET http://sefdb02.qut.edu.au:3001/volcano/16
Status
401 Unauthorized
Version
HTTP/1.1
Transferred
389 B (44 B size)
Referrer Policy
strict-origin-when-cross-origin
Response Headers (345 B)

9(a):


Volcano: Asamayama
Country: Japan
Region: Japan, Taiwan, Marianas
Subregion: Honshu
Last Eruption: 2019 CE
Summit: 2568
Elevation: 8425
BACK TO SEARCH
Pigeon | © OpenStreetMap contributors
Population
8,000,000
7,000,000
6,000,000
5,000,000
4,000,000
3,000,000
2,000,000
1,000,000
0
5km
10km
30km
100km

9(b):


Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFpbCI6Im1pa2VAZ21haWwuY29tIiwiZXhwIjoxNjUzOTE0MDE4LCJpYXQiOjE2NTM4Mjc2MTh9.hCE70b_rWtQli02wbW2Q4nCTCH6jnedhyPYN9Po8x3k